Требования к программам

1. Программа работает с двунаправленным списком объектов типа `student`:

```
class student
  {
  private:
    char * name  = nullptr;
    int    value = 0;
  public:
    ...
  };
class list2_node : public student
  {
  private:
    list2_node * next = nullptr;
    list2_node * prev = nullptr;
  public:
    ...
    friend class list2;
  };
class list2
  {
  private:
    list2_node * head = nullptr;
  public:
    ...
    int read (FILE *fp = stdin, unsigned int max_read = -1);
    void print (unsigned int r = 10, FILE *fp = stdout);
    unsigned int get_length ();
  };
```

   Все функции в задании являются членами класса "список".

2. Программа должна получать все параметры в качестве аргументов командной строки. Аргументы командной строки:

   1) r – количество выводимых значений в списке,
   2) `filename` – имя файла, откуда надо прочитать список.
   3) k – параметр для задачи.

   Например, запуск

```
./a.out 4 a.txt 2
```

   означает, что список прочитать из файла `a.txt`, выводить не более 4-х элементов списка и $k = 2$.

3. Класс "список" должен содержать функцию `read` ввода списка из указанного файла.

4. Ввод списка из файла. В указанном файле находится список в формате:

Слово-1	Целое-число-1
Слово-2	Целое-число-2
…	…
Слово-n	Целое-число-n

где слово – последовательность алфавитно-цифровых символов без пробелов. Концом ввода считается конец файла или достижение указанного значения `max_read` (в этом задании задается по умолчанию максимально возможным для типа `int`). Программа должна выводить сообщение об ошибке, если указанный файл не может быть прочитан или содержит данные неверного формата.

5. Решение задачи должно быть оформлено в виде подпрограммы, находящейся в отдельном файле. Получать в этой подпрограмме дополнительную информацию извне через глобальные переменные, включаемые файлы и т.п. запрещается.

6. Класс "список" должен содержать подпрограмму `print` вывода на экран списка длины не более r. Эта подпрограмма используется **для вывода исходного списка после его инициализации**, а также **для вывода на экран результирующего списка**. Подпрограмма выводит на экран не более, чем r элементов списка, где r – параметр этой подпрограммы (аргумент командной строки). Каждый элемент списка должен печататься на новой строке.

7. Класс "список" должен содержать функцию `get_length` вычисления длины списка. Эта функция используется для вывода длины исходного списка после его инициализации, а также для вывода на экран длины результирующего списка.

8. Вывод результата работы функции в функции `main` должен производиться по формату:

```
printf ("%s : Task = %d Len Old = %u Len New = %u Elapsed = %.2f\n",
        argv[0], task, len1, len2, t);
```

   где

   - `argv[0]` – первый аргумент командной строки (имя образа программы),
   - `task` – номер задачи (1–8),
   - `len1` – длина исходного списка,
   - `len2` – длина результирующего списка,
   - `t` – время работы функции, реализующей решение этой задачи.

   Вывод должен производиться в точности в таком формате, чтобы можно было автоматизировать обработку запуска многих тестов.

Задачи

1. Написать подпрограмму – член класса "список", получающую в качестве аргумента целое число k и осуществляющую циклический сдвиг элементов списка на k позиций вправо.

2. Написать подпрограмму – член класса "список", получающую в качестве аргумента целое число k и выбрасывающую из него все элементы, большие какого-либо из k предыдущих элементов (т.е. число x_i таких, что существует $j = 1,\ldots,k$ такое, что $x_i > x_{i-j}$).

3. Написать подпрограмму – член класса "список", получающую в качестве аргумента целое число k и выбрасывающую из него все элементы, большие какого-либо из k следующих элементов (т.е. число x_i таких, что существует $j = 1,\ldots,k$ такое, что $x_i > x_{i+j}$).

4. Написать подпрограмму – член класса "список", получающую в качестве аргумента целое число k и выбрасывающую из него все k-локальные максимумы (элемент x_i называется k-локальным максимумом, если $x_{i-k} \le \cdots \le x_{i-1} \le x_i \ge x_{i+1} \ge \cdots \ge x_{i+k}$ и для всех $j = 1,\ldots,k$ элементы x_{i-j}, x_{i+j} присутствуют в последовательности).

5. Написать подпрограмму – член класса "список", получающую в качестве аргумента целое число k и выбрасывающую из него все участки постоянства, имеющие длину больше k (участок удаляется целиком). Т.е. выбрасываются все наборы элементов $x_i = x_{i+1} = \cdots = x_{i+j}$ при $j \geq k$.

6. Написать подпрограмму – член класса "список", получающую в качестве аргумента целое число k и выбрасывающую из него все участки нестрого убывания, имеющие длину больше k (участок удаляется целиком). Т.е. выбрасываются все наборы элементов $x_i \geq x_{i+1} \geq \cdots \geq x_{i+j}$ при $j \geq k$.

7. Написать подпрограмму – член класса "список", получающую в качестве аргумента целое число k и выбрасывающую из него все участки между участками постоянства, имеющими длину больше k (участок удаляется целиком).

8. Написать подпрограмму – член класса "список", получающую в качестве аргумента целое число k и выбрасывающую из него все участки между участками нестрого возрастания, имеющие длину больше k (участок удаляется целиком).